

**JP8297567**

Publication Title:

DECENTRALIZED APPLICATION DEVELOPMENT SUPPORT DEVICE

Abstract:

Abstract of JP8297567

**PURPOSE:** To facilitate the arrangement and installation of application constituent elements of a decentralized application. **CONSTITUTION:** A physical configuration automatic generating means 72 automatically generates a physical configuration as the arrangement of the application constituent elements in physical execution environment from logical execution environment definitions, a logical configuration, execution environment mapping, and physical execution environment definitions. An execution module generating means 40 generates execution modules by arrangement destinations according to the physical configuration. The arrangement of the application constituent elements is done not in a physical execution environment, but in a logical execution environment and then the arrangement of the application constituent elements becomes higher in independence of the physical execution environment to decrease the man-hours needed for application to plural different physical execution environments and application to a physical execution environment changing in constitution.

Data supplied from the esp@cenet database - Worldwide

-----  
Courtesy of <http://v3.espacenet.com>

(19)日本国特許庁 (J P)

(12) 公 開 特 許 公 報 (A)

(11)特許出願公開番号

特開平8-297567

(43)公開日 平成8年(1996)11月12日

(51)Int.Cl.<sup>6</sup>

G 0 6 F 9/06  
15/16

識別記号

4 1 0  
4 3 0

庁内整理番号

F I

G 0 6 F 9/06  
15/16

技術表示箇所

4 1 0 B  
4 3 0 A

審査請求 未請求 請求項の数7 O L (全 19 頁)

(21)出願番号 特願平7-102406

(22)出願日 平成7年(1995)4月26日

(71)出願人 000006013

三菱電機株式会社  
東京都千代田区丸の内二丁目2番3号

(72)発明者 原田 道明

鎌倉市大船五丁目1番1号 三菱電機株式  
会社情報システム研究所内

(74)代理人 弁理士 宮田 金雄 (外3名)

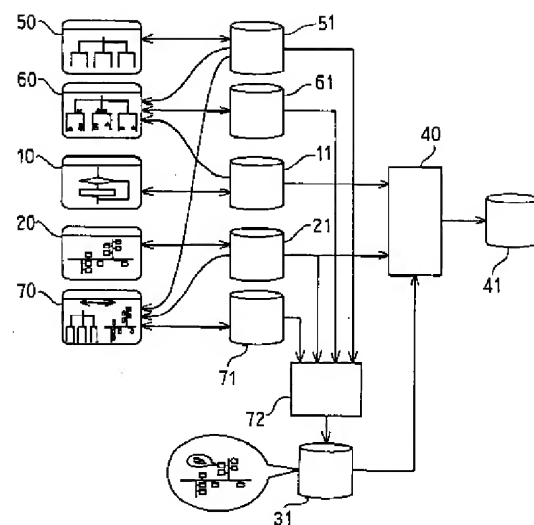
(54)【発明の名称】 分散アプリケーション開発支援装置

(57)【要約】 (修正有)

【目的】 分散アプリケーションのアプリケーション構成要素の配置、インストールを容易にする。

【構成】 物理コンフィグレーション自動生成手段72は論理実行環境定義、論理コンフィグレーション、実行環境マッピング、物理実行環境定義から物理実行環境上のアプリケーション構成要素の配置である物理コンフィグレーションを自動生成する。実行モジュール生成手段40は物理コンフィグレーションに従い、配置先別の実行モジュールを生成する。

【効果】 アプリケーション構成要素の配置を物理実行環境ではなく論理実行環境上で行うことにより、アプリケーション構成要素の配置が物理実行環境と独立性の高いものになり、異なる複数の物理実行環境への対応や構成の変動する物理実行環境への対応に必要な工数を削減することができる。



10:アプリケーション構成要素定義手段 50:論理実行環境定義手段  
11:アプリケーション構成要素記憶手段 51:論理実行環境記憶手段  
20:物理実行環境定義手段 60:論理コンフィグレーション編集手段  
21:物理実行環境記憶手段 61:論理コンフィグレーション記憶手段  
31:物理コンフィグレーション記憶手段 70:実行環境マッピング編集手段  
40:実行モジュール生成手段 71:実行環境マッピング記憶手段  
41:実行モジュール記憶手段 72:物理コンフィグレーション自動生成手段

## 【特許請求の範囲】

【請求項1】 下記的手段を備えたことを特徴とする分散アプリケーション開発支援装置。

(a) アプリケーション構成要素を作成するアプリケーション構成要素定義手段、

(b) 上記アプリケーション構成要素定義手段により作成されたアプリケーション構成要素を格納するアプリケーション構成要素記憶手段、

(c) 論理的な計算資源を指定するための仮想計算資源である論理リソースを用いて、上記論理リソースの集合によりモデル化して規定され論理実行環境定義情報を作成する論理実行環境編集手段、

(d) 上記論理実行環境編集手段により作成された論理実行環境定義情報を格納する論理実行環境記憶手段、

(e) 上記論理実行環境記憶手段に格納された論理実行環境定義情報を選択して表示し、該表示上で上記アプリケーション構成要素記憶手段に格納されたアプリケーション構成要素の配置を指定することにより、アプリケーションを構成する各要素の論理リソースへの配置情報を規定した論理コンフィグレーション情報を作成する論理コンフィグレーション編集手段、

(f) 上記論理コンフィグレーション編集手段により作成された論理コンフィグレーション情報を格納する論理コンフィグレーション記憶手段、

(g) アプリケーションの実行に使用される計算資源の集合である物理実行環境定義情報を、物理実行環境を構成する個々の計算資源である物理リソース定義情報を用いて作成する物理実行環境編集手段、

(h) 上記物理実行環境編集手段により作成された物理実行環境定義情報を格納する物理実行環境記憶手段、

(i) 上記論理実行環境記憶手段に格納された論理実行環境定義情報と、物理実行環境記憶手段に格納された物理実行環境定義情報を選択して表示し、該表示上で論理実行環境定義情報に含まれる論理リソースと物理実行環境定義情報に含まれる物理リソースの対応づけを規定するための実行環境マッピング情報を作成する実行環境マッピング編集手段、

(j) 上記実行環境マッピング編集手段により作成された実行環境マッピング情報を格納する実行環境マッピング記憶手段、

(k) 上記論理実行環境記憶手段に格納された論理実行環境定義情報と、上記論理コンフィグレーション記憶手段に格納された論理コンフィグレーション情報と、上記実行環境マッピング記憶手段に格納された実行環境マッピング情報と、上記物理実行環境記憶手段に格納された物理実行環境定義情報に基づいて、アプリケーション構成要素の物理リソースに対する配置を規定した物理コンフィグレーション情報を自動生成する物理コンフィグレーション自動生成手段、

(l) 上記物理コンフィグレーション自動生成手段によ

り生成された物理コンフィグレーション情報を格納する物理コンフィグレーション記憶手段、

(m) 上記物理コンフィグレーション記憶手段に格納された物理コンフィグレーション情報に基づいて上記アプリケーション構成要素記憶手段に格納されたアプリケーション構成要素をコンパイル・リンクし、実行モジュールの生成を行う実行モジュール生成手段、

(n) 上記実行モジュール生成手段により生成された実行モジュールを格納する実行モジュール記憶手段。

10 【請求項2】 上記実行モジュール記憶手段に格納された実行モジュールを配布・インストールするアプリケーションインストール手段を備えたことを特徴とする請求項1記載の分散アプリケーション開発支援装置。

【請求項3】 下記的手段を備えたことを特徴とする請求項2記載の分散アプリケーション開発支援装置。

(a) 上記論理実行環境記憶手段に格納された論理実行環境定義情報を表示し、論理実行環境上においてサービスを透過的に呼び出すことのできる領域を論理ドメインとして選択し、該論理ドメインに対し識別名を与えることによって論理ドメインを定義する論理ドメイン編集手段、

(b) 上記論理ドメイン編集手段によって定義された論理ドメイン情報を格納する論理ドメイン記憶手段、

(c) 上記論理ドメイン記憶手段に格納された論理ドメイン情報と上記実行環境マッピング記憶手段に格納された実行環境マッピング情報とを用いてサービスを透過的に呼び出すことのできる物理実行環境上のドメインを物理ドメインとして算出する物理ドメイン算出手段、

30 (d) 上記物理ドメイン算出手段の算出した物理ドメイン情報に基づいて物理実行環境に備えられた等価性制御機構に対してドメインの設定を行う物理ドメイン設定手段。

【請求項4】 上記物理コンフィグレーション記憶手段に格納された物理コンフィグレーション情報を編集するための物理コンフィグレーション編集手段を備えたことを特徴とする請求項第1項乃至第3項いずれかに記載の分散アプリケーション開発支援装置。

40 【請求項5】 上記物理コンフィグレーション編集手段は、上記物理コンフィグレーション編集手段によって編集された物理コンフィグレーション情報が該物理コンフィグレーションの自動生成の入力に用いられた論理コンフィグレーション情報および実行環境マッピング情報と矛盾しているか否かを検査する物理コンフィグレーション一貫性検査手段を備えたことを特徴とする請求項第1項乃至第4項いずれかに記載の分散アプリケーション開発支援装置。

【請求項6】 論理実行環境定義を木構造により階層化して構成し、該木構造を構成する各節点は論理リソースの集合と対応づけて表現するようにしたことを特徴とする請求項第1項乃至第5項いずれかに記載の分散アプリ

ケーション開発支援装置。

【請求項7】 論理ドメインは論理実行環境を構成する木構造の節点と一対一に対応付けられることを特徴とする請求項6記載の分散アプリケーション開発支援装置。

【発明の詳細な説明】

【0001】

【産業上の利用分野】 この発明は、計算機ネットワーク上に処理を分散した計算機アプリケーションの構築を支援する分散アプリケーション開発支援装置に関する。

【0002】

【従来の技術】 従来の技術を説明する前に、まず必要な用語について説明する。

(1) 物理実行環境、物理実行環境定義

アプリケーションの実行に使用される物理的な計算資源の構成を物理実行環境と定義する。物理実行環境定義はアプリケーション開発者が分散アプリケーション開発支援装置に対して物理実行環境を定義するための情報であり、下記の物理リソースの集合によって表現される。また、必要に応じて、物理実行環境定義に計算機ネットワークの構成、計算機ネットワーク上の物理リソースの配置を追加することがある。

(2) 物理リソース、物理リソース定義

アプリケーションの実行に使用される実在の計算資源を物理リソースと定義する。物理リソース定義は、分散アプリケーション開発支援装置に対して物理リソースを定義するための情報であり、以下の情報を含む。

- a. 資源の種別（計算機、あるいはデータベースサーバ）
- b. 計算機である物理リソースに対して、ノード名および機種情報（CPUの種別、OSの種別等）
- c. データベースサーバである物理リソースに対する機種情報、および接続情報

(3) 物理コンフィグレーション

アプリケーション、あるいはアプリケーションの部分構成要素の物理リソースへの配置を定義した情報を、該アプリケーションの物理コンフィグレーションと定義する。

(4) 論理実行環境、論理実行環境定義

論理実行環境は論理的な計算資源の構成を示すことを目的として仮想的な計算資源を導入し、アプリケーションの実行に使用される計算資源構成を示したモデルを論理実行環境と定義する。論理実行環境定義は分散アプリケーション開発支援装置に対して論理実行環境の定義を与えるもので、下記の論理リソースの集合により表現される。また、必要に応じて、論理実行環境定義に計算機ネットワークの構成と、計算機ネットワーク上の論理リソースの配置を含める事ができる。

(5) 論理リソース、論理リソース定義

論理実行環境を構成する仮想的な計算資源であり、計算機あるいはデータベースサーバに対応づけられる。論理

リソース定義は分散アプリケーション開発支援装置に対して論理リソースの定義を与える情報で、以下の情報を含む。

a. 資源の種別

b. 識別名

(6) 論理コンフィグレーション

アプリケーション、あるいはアプリケーションの部分構成要素の論理実行環境上の計算資源への配置を定義した情報を、該アプリケーションの論理コンフィグレーションと定義する。

(7) サービス

アプリケーション構成要素が他のアプリケーション構成要素に対して公開する機能呼出インタフェースを、サービスと定義する。

(8) 透過性制御機構

計算機ネットワークシステムでは、アプリケーション構成要素からのサービスの呼び出し要求に対し、サービスを呼び出す側とサービスを提供する側のアプリケーション構成要素との関係に応じて、呼び出し要求を特定のアプリケーション構成要素の実体へと導く機構が用意される。この機構を透過性制御機構と定義する。透過性制御機構の実現方式としては、例えば「名前サービス」とよばれる機構が広く利用されている。

(9) ドメイン

計算環境上に配置されたサービスは、様々な計算資源上に配置されたアプリケーション構成要素から呼び出すことができるが、運用上、特定の位置に置かれたサービスは特定の計算資源の集合からのみ呼び出せるように透過性制御機構を用いて制御することが一般的に行われている。このため、サービスの提供範囲を、計算資源の集合を単位として制御する方式が広く用いられる。この単位となる計算資源の集合をドメインと定義する。また、特定のサービスに対して、該サービスを透過的に呼び出せる計算環境上の計算資源の集合を、そのサービスのドメインと呼ぶ。

(10) 論理ドメイン

論理実行環境上において定義されるドメインを、論理ドメインと定義する。論理ドメインは、論理リソースの集合である。

(11) 物理ドメイン

物理実行環境上において定義されるドメインを、物理ドメインと定義する。物理ドメインは、物理リソースの集合である。

【0003】 従来の技術における分散アプリケーション開発支援装置について、図18乃至図20に基づいて説明する。図18は、従来の分散アプリケーション開発支援装置におけるアプリケーション構成要素配置指定方式の例として、米国Forte Software Inc.社の開発した分散アプリケーション開発支援装置Forteのアプリケーション構成要素配置指定方式

に関する部分の構成を示した概念図である (Forte マニュアル「A Guide to the Forte Workshops」Version 1.0, June 30, 1994. Forte Software Inc.)。図中、10 (Forte の「Class Workshop」「Method Workshop」「Window Workshop」等に相当する) はアプリケーション構成要素を作成するアプリケーション構成要素定義手段、11 はアプリケーション構成要素定義手段10によって作成されたアプリケーション構成要素を格納するアプリケーション構成要素記憶手段、20 (Forte の「Environment Workshop」に相当する) は物理実行環境定義を作成する物理実行環境編集手段、21 は物理実行環境編集手段20によって作成された物理実行環境定義情報を格納する物理実行環境記憶手段である。また、30 (Forte マニュアルでは「Partition Workshop」に相当する) は物理コンフィグレーションを作成する物理コンフィグレーション編集手段、31 は作成された物理コンフィグレーション情報を格納する物理コンフィグレーション記憶手段、40 は物理コンフィグレーション記憶手段31に格納された物理コンフィグレーション情報の内容に基づいてアプリケーション構成要素を適切にコンパイル・リンクし、各物理リソースに配布すべき実行モジュールを生成する実行モジュール生成手段である。また、41 は生成された実行モジュールを格納する実行モジュール記憶手段、42 は実行モジュールを各物理リソースに配布し、インストールするアプリケーションインストール手段である。

【0004】次に、アプリケーション構成要素配置指定方法の動作について説明する。従来の分散アプリケーション開発支援装置においては、アプリケーション開発者は図19に示す順序で分散アプリケーションの開発を行っていた。アプリケーション開発者はアプリケーション構成要素定義手段10を使用してアプリケーション構成要素を作成し、その結果はアプリケーション構成要素記憶手段11に格納される (S101)。また、アプリケーション開発者は物理実行環境編集手段20を使用して物理実行環境定義を作成し、その結果を物理実行環境記憶手段21に格納する (S102)。

【0005】物理実行環境定義はアプリケーションと独立に作成できる。また、すでに作成済みの物理実行環境を別のアプリケーションの作成にも利用することもできる。さらに、同一のアプリケーションに対して異なる物理実行環境を作成し、複数の異なった実行環境に対してアプリケーション構成要素の配置を同時に設計することも可能である。

【0006】次にアプリケーション開発者は、物理コンフィグレーション編集手段30を使用して物理コンフィグレーション情報を作成する (S103)。物理コンフ

ィグレーション編集手段は図20で示すように物理実行環境を表示し、アプリケーション開発者は表示された物理リソース上 (Forte の例では物理リソースとして計算機ネットワークの1ノードを指定する) に対するアプリケーション構成要素の配置を指定する。物理コンフィグレーション情報は、物理コンフィグレーション記憶手段31に格納される。

【0007】1つの物理コンフィグレーションは、ある与えられた1つのアプリケーションと1つの物理実行環境の組に対して、可能なアプリケーション構成要素の配置の1つを定義するものである。Forte は、アプリケーションと物理実行環境の1組に対して複数の物理コンフィグレーションを定義することができる。これによって様々なアプリケーション構成要素の配置を容易に試みることができ、アプリケーション開発者が最適な配置を容易に見出すことができる。

【0008】実行モジュール生成手段40は、各物理リソースに配置すべきアプリケーション構成要素の集合を物理コンフィグレーション記憶手段31に格納された物理コンフィグレーションから算出し、コンパイル・リンクを行って実行モジュールを生成する (S104)。

【0009】このとき、実行モジュール生成手段40は物理実行環境定義に含まれる物理リソース定義を参照して物理リソースの特性 (機種情報等) を取得し、各アプリケーション構成要素を配置先の物理リソースのもつ特性と合致するようにコンパイルし、リンクする。

【0010】アプリケーションインストール手段42は、実行モジュール生成手段40により生成された実行モジュールを各物理リソースに配布・インストールする (S105~S106)。

【0011】

【発明が解決しようとする課題】従来の分散アプリケーション開発支援装置では以上のようにしてアプリケーション構成要素の配置指定を実現しており、アプリケーション開発者は物理コンフィグレーションによってアプリケーション構成要素の物理リソースに対する配置を直接に指定する。

【0012】このため、物理実行環境が変化する都度、アプリケーション開発者は変化の生じた物理リソースに関連するアプリケーション構成要素の配置を全て修正する必要があった。また、アプリケーション開発者は物理実行環境上で実行される全てのアプリケーションの物理コンフィグレーション情報を修正しなければならず、多数のアプリケーションから構成される大規模システムを構築、維持して行くためには多大の作業工数を要するという問題点があった。

【0013】また、アプリケーション構成要素の提供する個々のサービスに対してサービスを透過的に呼び出せる実行環境上の領域 (ドメイン) を陽に定義する機能を備えていなかったため、システム管理者が実行環境上に

配置された各サービスに対するドメインの設定を逐一手動で設定しなければならず、特に大規模な分散アプリケーションシステムにおいては多くの作業量を必要とするという問題点があった。

【0014】この発明は上記のような問題点を解消するためになされたもので、物理的実行環境の変化に応じ、分散システムを構成するアプリケーション構成要素に対するシステム再構築のための作業を容易、且つ迅速に行えるようにすることを目的とする。

【0015】さらに、アプリケーション構成要素の透過性制御に関する設定を容易に行い、物理実行環境の変動に対して独立性の高いものとするることにより、分散システムの構築に対し柔軟に対処できるようにすることを目的とする。

【0016】

【課題を解決するための手段】第1の発明に係る分散アプリケーション開発支援装置は、以下の構成要素を備えるようにしたものである。

(a) アプリケーション構成要素を作成するアプリケーション構成要素定義手段、(b) 上記アプリケーション構成要素定義手段により作成されたアプリケーション構成要素を格納するアプリケーション構成要素記憶手段、

(c) 論理的な計算資源を指定するための仮想計算資源である論理リソースを用いて、上記論理リソースの集合によりモデル化して規定され論理実行環境定義情報を作成する論理実行環境編集手段、(d) 上記論理実行環境編集手段により作成された論理実行環境定義情報を格納する論理実行環境記憶手段、(e) 上記論理実行環境記憶手段に格納された論理実行環境定義情報を選択して表示し、該表示上で上記アプリケーション構成要素記憶手段に格納されたアプリケーション構成要素の配置を指定することにより、アプリケーションを構成する各要素の論理リソースへの配置情報を規定した論理コンフィグレーション情報を作成する論理コンフィグレーション編集手段、(f) 上記論理コンフィグレーション編集手段により作成された論理コンフィグレーション情報を格納する論理コンフィグレーション記憶手段、(g) アプリケーションの実行に使用される計算資源の集合である物理実行環境定義情報を、物理実行環境を構成する個々の計算資源である物理リソース定義情報を用いて作成する物理実行環境編集手段、(h) 上記物理実行環境編集手段により作成された物理実行環境定義情報を格納する物理実行環境記憶手段、(i) 上記論理実行環境記憶手段に格納された論理実行環境定義情報と、物理実行環境記憶手段に格納された物理実行環境定義情報を選択して表示し、該表示上で論理実行環境定義情報に含まれる論理リソースと物理実行環境定義情報に含まれる物理リソースの対応づけを規定するための実行環境マッピング情報を作成する実行環境マッピング編集手段、(j) 上記実行環境マッピング編集手段により作成された実行環境マッ

ピング情報を格納する実行環境マッピング記憶手段、

(k) 上記論理実行環境記憶手段に格納された論理実行環境定義情報と、上記論理コンフィグレーション記憶手段に格納された論理コンフィグレーション情報と、上記実行環境マッピング記憶手段に格納された実行環境マッピング情報と、上記物理実行環境記憶手段に格納された物理実行環境定義情報に基づいて、アプリケーション構成要素の物理リソースに対する配置を規定した物理コンフィグレーション情報を自動生成する物理コンフィグレーション自動生成手段、(l) 上記物理コンフィグレーション自動生成手段により生成された物理コンフィグレーション情報を格納する物理コンフィグレーション記憶手段、(m) 上記物理コンフィグレーション記憶手段に格納された物理コンフィグレーション情報に基づいて上記アプリケーション構成要素記憶手段に格納されたアプリケーション構成要素をコンパイル・リンクし、実行モジュールの生成を行う実行モジュール生成手段、(n) 上記実行モジュール生成手段により生成された実行モジュールを格納する実行モジュール記憶手段。

【0017】第2の発明は、第1の発明に係る分散アプリケーション開発支援装置において、実行モジュール記憶手段に格納された実行モジュールを配布・インストールするアプリケーションインストール手段を備えるようにしたものである。

【0018】第3の発明は、第2の発明に係る分散アプリケーション開発支援装置において、以下の手段を備えるようにしたものである。

(a) 上記論理実行環境記憶手段に格納された論理実行環境定義情報を表示し、論理実行環境上においてサービスを透過的に呼び出すことのできる領域を論理ドメインとして選択し、該論理ドメインに対し識別名を与えることによって論理ドメインを定義する論理ドメイン編集手段、(b) 上記論理ドメイン編集手段によって定義された論理ドメイン情報を格納する論理ドメイン記憶手段、

(c) 上記論理ドメイン記憶手段に格納された論理ドメイン情報と上記実行環境マッピング記憶手段に格納された実行環境マッピング情報とを用いてサービスを透過的に呼び出すことのできる物理実行環境上のドメインを物理ドメインとして算出する物理ドメイン算出手段、

(d) 上記物理ドメイン算出手段の算出した物理ドメイン情報に基づいて物理実行環境に備えられた等価性制御機構に対してドメインの設定を行う物理ドメイン設定手段。

【0019】第4の発明は、第1乃至第3の発明に係る分散アプリケーション開発支援装置において、物理コンフィグレーション記憶手段に格納された物理コンフィグレーション情報を編集するための物理コンフィグレーション編集手段を備えるようにしたものである。

【0020】第5の発明は、第1乃至第4の発明に係る分散アプリケーション開発支援装置において、物理コン

フィグレーション編集手段によって編集された物理コンフィグレーション情報が、該物理コンフィグレーションの自動生成の入力に用いられた論理コンフィグレーション情報および実行環境マッピング情報と矛盾しているか否かを検査する物理コンフィグレーション一貫性検査手段を備えるようにしたものである。

【0021】第6の発明は、第1乃至第5の発明に係る分散アプリケーション開発支援装置において、論理実行環境定義を木構造により階層化して構成し、木構造を構成する各節点を論理リソースの集合と対応づけて表現するようにしたものである。

【0022】第7の発明は、第6の発明に係る分散アプリケーション開発支援装置において、論理ドメインを論理実行環境を構成する木構造の節点と一対一に対応付けるようにして構成したものである。

【0023】

【作用】第1の発明による分散アプリケーション開発支援装置では、アプリケーション開発者はまず、アプリケーション構成要素定義手段を用いてアプリケーション構成要素を、また、論理実行環境編集手段を用いて論理実行環境定義を作成する。作成されたアプリケーション構成要素、および論理実行環境定義は、各々アプリケーション構成要素記憶手段、論理実行環境記憶手段に格納される。

【0024】次にアプリケーション開発者は、論理コンフィグレーション編集手段を用いてアプリケーション構成要素の論理実行環境上の配置である論理コンフィグレーションを作成する。このとき、論理コンフィグレーション編集手段は、上記論理実行環境記憶手段に格納された論理実行環境定義を表示し、上記アプリケーション構成要素記憶手段に格納されたアプリケーション構成要素を論理リソースの上にユーザ操作により配置することで論理コンフィグレーションを作成する。作成された論理コンフィグレーションは、論理コンフィグレーション記憶手段に格納される。

【0025】また、アプリケーション開発者は、物理実行環境編集手段を用いて物理実行環境定義を作成する。作成された物理実行環境定義は物理実行環境記憶手段に格納される。

【0026】また、アプリケーション開発者は、実行環境マッピング編集手段を使用して実行環境マッピング情報を作成する。実行環境マッピング編集手段は上記論理実行環境記憶手段に格納された論理実行環境定義、物理実行環境記憶手段に格納された物理実行環境定義に含まれる論理リソース、物理リソースの集合を表示し、アプリケーション開発者がこれらの対応関係を指定することにより実行環境マッピング情報が作成される。作成された実行環境マッピング情報は実行環境マッピング記憶手段に格納される。

【0027】物理コンフィグレーション自動生成手段

は、アプリケーション開発者により指定された論理実行環境情報、論理コンフィグレーション情報、物理実行環境情報、実行環境マッピング情報をそれぞれ論理実行環境記憶手段、論理コンフィグレーション記憶手段、物理実行環境記憶手段、実行環境マッピング記憶手段から入力して、物理コンフィグレーションを自動生成する。生成された物理コンフィグレーション情報は物理コンフィグレーション記憶手段に格納される。

【0028】実行モジュール生成手段は、上記により生成された物理コンフィグレーション情報を使用して実行モジュールの生成を行う。

【0029】また第2の発明は、第1の発明による分散アプリケーション開発支援装置において実行モジュール生成手段が生成した実行モジュールを、アプリケーションインストール手段により配布、インストールする。

【0030】また、第3の発明は第2の発明に係る分散アプリケーション開発支援装置において、アプリケーション開発者は論理ドメイン編集手段を用いて論理ドメイン定義を行なう。このとき、論理ドメイン編集手段は上記論理実行環境記憶手段に格納された論理実行環境定義情報を表示し、ユーザ操作によって指定された論理リソースの集合に識別名を与えて1つの論理ドメインとして定義する。作成された論理ドメイン定義は論理ドメイン記憶手段に格納される。論理コンフィグレーション編集手段は各論理リソースに配置されたアプリケーション構成要素に対し、アプリケーション構成要素の提供する各サービスについて、サービスの提供先となる論理ドメインの識別名をアプリケーション開発者に指定させる。一つのサービスに対して複数の論理ドメインを指定することが可能である。該指定は論理コンフィグレーションの一部として論理コンフィグレーション記憶手段に格納される。物理ドメイン算出手段は、論理ドメイン記憶手段に格納された論理ドメイン情報と実行環境マッピング記憶手段に格納された実行環境マッピング情報とを用いて物理ドメインを算出する。物理ドメイン設定手段は、物理ドメイン算出手段により算出された物理ドメイン情報に基づき、物理実行環境に備えられた透過性制御手段に対して、透過性制御を指示する。

【0031】第4の発明は第1乃至第3の発明に係る分散アプリケーション開発支援装置において、物理コンフィグレーション編集手段を用いて物理コンフィグレーション自動生成手段により自動生成された物理コンフィグレーション情報を修正し、最適化を図る。

【0032】第5の発明は第1乃至第4の発明に係る分散アプリケーション開発支援装置において、物理コンフィグレーション一貫性検査手段により物理コンフィグレーション編集手段で編集された物理コンフィグレーション情報が該物理コンフィグレーションの自動生成の入力に用いられた論理コンフィグレーション情報および実行環境マッピング情報と矛盾しているかどうかを検査す



る。

【0033】第6の発明は第1乃至第5の発明に係る分散アプリケーション開発支援装置において、論理実行環境を木構造を用いた階層構造とし、木構造の各節点はこれと対応付けられた論理リソースの集合として表現する。

【0034】第7の発明は第6の発明に係る分散アプリケーション開発支援装置において、論理ドメインを木構造の特定の1節点により表現し、論理ドメインは指定された特定の節点を根とする部分木全体を構成する節点と

【0035】

【実施例】

実施例1. 本発明の第1の実施例を、図1乃至図7に基づいて説明する。図1は分散アプリケーション開発支援装置の構成図であり、図において、10、11、20、21、31、40、41は従来装置と同等の構成要素を示す。また、図2は第1の実施例におけるアプリケーション開発者のアプリケーション開発手順を示したものである。

【0036】アプリケーション開発者は、従来と同様にしてアプリケーション構成要素を作成し(S201)、また、論理実行環境編集手段50を使用して論理実行環境定義を作成する(S202)。作成された論理実行環境定義は論理実行環境記憶手段51に格納される。このとき論理実行環境編集手段50は図3に示すような表示を用いて、論理実行環境を構成する各論理リソースを定義する。図3の例では、まず論理的な広域計算機ネットワークWAN1、局所計算機ネットワークLAN1、LAN2の接続を定義する。次に定義された計算機ネットワーク上に各論理リソースを配置し、また論理リソースの名称、種別を定義する。論理的な計算機ネットワークを定義する機能は本発明の実施例において必ずしも必須の機能ではなく、論理実行環境編集手段は単に個々の論理リソースの定義の集まりを作成する手段であってもよい。しかしながら、この実施例で示すように論理的な計算機ネットワークを仮定して、その上に論理リソースを配置する方式をとれば、アプリケーション開発者は実際の計算機ネットワーク上の物理リソースとの対応をもとに論理リソースの役割を容易に理解することができ、作業効率を高めることができる。

【0037】次に、アプリケーション開発者は、論理コンフィグレーション編集手段60を用いて論理コンフィグレーションを作成する(S203)。

【0038】このとき論理コンフィグレーション編集手段60は、例えば図4に示すようにして論理実行環境記憶手段51に格納された論理実行環境定義を表示する。アプリケーション開発者は、該表示上でアプリケーション構成要素記憶手段11に格納されたアプリケーション構成要素を論理リソースの上に配置することによって論

理コンフィグレーションを編集する。図4の例では、論理リソース(node3)上にモジュールA、B、Cから成るアプリケーション構成要素が配置された様子が示されている。作成された論理コンフィグレーションは論理コンフィグレーション記憶手段61に格納される。

【0039】また、アプリケーション開発者は物理実行環境編集手段20を用いて物理実行環境定義を作成する(S204)。作成された物理実行環境定義は物理実行環境記憶手段21に格納される。この時、物理実行環境編集手段20は図5に示すような表示を用いて、物理実行環境を構成する各物理リソースと計算機ネットワークを定義する。図5の例では、選択された物理リソース(host3)に対する物理リソース特性として、種別、名称、機種、OS、OSバージョンを定義している様子が示されている。

【0040】また、アプリケーション開発者は論理実行環境記憶手段51と、物理実行環境記憶手段21に格納された論理実行環境定義、および物理実行環境定義に対して、論理リソースと物理リソースの間の対応関係を定義した実行環境マッピング情報を実行環境マッピング編集手段70を使用して作成する(S205)。作成された実行環境マッピング情報は実行環境マッピング記憶手段71に格納される。図6に、論理リソースと物理リソースを多対多に関連付けた実行環境マッピング情報の例を示す。

【0041】物理コンフィグレーション自動生成手段72は、アプリケーション開発者により指定された論理実行環境情報、論理コンフィグレーション情報、物理実行環境情報、実行環境マッピング情報をそれぞれ論理実行環境記憶手段51、論理コンフィグレーション記憶手段61、物理実行環境記憶手段21、実行環境マッピング記憶手段71から入力して、物理コンフィグレーションを自動生成する(S206)。生成された物理コンフィグレーションは物理コンフィグレーション記憶手段31に格納される。ここで、物理コンフィグレーションは以下の条件を満たす必要がある。

(a) ある論理リソースに配置されたアプリケーション構成要素は、関連付けられた物理リソースの少なくとも1つに配置される。

(b) 複数箇所に重複して配置すると正しい動作が保証されないアプリケーション構成要素は、関連付けられた物理リソースの何れか1つのみに配置される。  
上記の条件を満たすアルゴリズムとして、例えば、図7に示すアルゴリズムを使用することができる。

【0042】実行モジュール生成手段40は、上記により生成された物理コンフィグレーションを使用して従来技術と同様にして実行モジュールの生成を行う(S207)。

【0043】なお、第1の実施例では、図18に示した物理コンフィグレーション編集手段30、およびアプリ



ケーションインストール手段42は必須要素ではないので、これらを省略した構成とした。アプリケーションインストール手段42を省略した場合は、アプリケーションの配布、実行管理、環境設定はアプリケーション開発者自身が行えばよい。また、物理コンフィグレーションは従来の実施例とは異なり自動生成される為、物理コンフィグレーション編集手段30は必ずしも必要ではない。

【0044】第1の実施例によれば、物理実行環境における計算資源の構成が変動しても論理コンフィグレーションの修正は不要であり、物理実行環境および実行環境マッピングを修正するだけでアプリケーション構成要素を再配置することができるので、システムの構築、維持を容易に行うことができる。

【0045】実施例2. この発明の第2の実施例を、図8について説明する。第1の実施例では従来装置におけるアプリケーションインストール手段42を省略した構成を示したが、図8に示すように、アプリケーションインストール手段42を付加することにより、実行モジュール記憶手段41に格納された実行モジュールの配布、インストール処理を該アプリケーションインストール手段42を用いて実現してもよい。これにより、アプリケーションの配布、インストール作業を容易に実施することが可能となる。

【0046】実施例3. この発明の第3の実施例を、図9乃至図12に基づいて説明する。本実施例は第2の実施例における分散アプリケーション開発支援装置に対して、論理実行環境上のドメインの定義である論理ドメイン定義を編集する論理ドメイン編集手段80と、作成された論理ドメイン定義を格納する論理ドメイン記憶手段81を備え、さらに論理ドメインと実行環境マッピングから物理ドメインを算出する物理ドメイン算出手段43と、アプリケーションインストール手段42の内部に物理ドメイン設定手段44を備えるようにしたものである。

【0047】アプリケーション開発者は、図10に示すように論理ドメイン編集手段80を用いて論理ドメイン定義を作成する。論理ドメイン編集手段80は論理実行環境記憶手段51に格納された論理実行環境定義を表示し、ユーザ操作によって指定された論理リソースの集合を1つの論理ドメインとして定義する。作成された論理ドメイン定義は論理ドメイン記憶手段81に格納される。図10では、node2とnode3から構成される論理リソースの集合に対して”鎌倉地区倉庫”と名前を付け、新しい論理ドメインとして定義する様子を示している。

【0048】論理コンフィグレーション編集手段60は論理コンフィグレーション定義として、第1の実施例で用いられる論理コンフィグレーションに加えて、更に、論理リソース上に配置された各アプリケーション構成要

素の各サービスに対し該サービスを透過的に呼び出すことが可能な論理ドメインを、アプリケーション開発者に指定させる。この時、論理コンフィグレーション編集手段60は図11に示すように、アプリケーション開発者に各論理リソースに配置されたアプリケーション構成要素のサービスを選択させ、選択されたサービスがどの論理ドメインから透過的に呼び出すことができるかをアプリケーション開発者に指定させる。図11では、node3で指定された論理リソース上には、サービス一覧として”受注”、”在庫管理”、”入金”が登録されており、”在庫管理”なるサービスは”支社営業”、”販売店”、”地区倉庫”の各ドメインから利用可能であることを示している。論理コンフィグレーション記憶手段61は第一の実施例に用いられる論理コンフィグレーション情報に加えて、上記の指定情報を格納する。

【0049】物理ドメイン算出手段43は論理ドメインと実行環境マッピング情報から物理ドメインを算出する。例えば、物理ドメイン算出手段43は図12に示すフローチャートによって、物理ドメインを算出することができる。アプリケーションインストール手段42内に設けられた物理ドメイン設定手段44は、この算出結果に基づいて、実行環境に備えられた透過性制御手段に対し、物理実行環境上に配置された各サービスの物理的な位置、および各サービス呼び出すことのできる物理ドメインを知らせる。これにより、サービスの透過性に関する設定を自動的に行うことができる。

【0050】この実施例によれば、アプリケーション開発者は各サービスの提供範囲をドメインとして指定し、この指定情報に基づいて物理実行環境に備えられた透過性制御手段をコントロールするようにしたので、各サービスに対する透過性を正しく制御することができる。また、独立性の高い論理ドメイン定義情報を用いるようにしたので、物理実行環境の変動が発生しても、システムにおけるサービスの透過性に対する設定、変更作業を容易に行うことができる。

【0051】実施例4. この発明の第4の実施例を、図13に基づいて説明する。第4の実施例は、第1の実施例における分散アプリケーション開発支援装置に対し、さらに物理コンフィグレーション自動生成手段72の生成した物理コンフィグレーションをアプリケーション開発者が編集するための物理コンフィグレーション編集手段30を備える。物理コンフィグレーション編集手段30としては、従来技術によるものと全く同じものを使用することができる。

【0052】本実施例によれば、物理コンフィグレーション自動生成手段を用いて自動生成された物理コンフィグレーション情報に対し、さらに物理実行環境に合わせてアプリケーション構成要素をきめ細かく最適化して配置することができるので、アプリケーションの実行効率を高めることが可能となる。また、物理実行環境の変化

によるシステムの維持、保全に対しても、少ない作業量で柔軟かつ容易に対処することができる。

【0053】実施例5. この発明の第5の実施例を図14、および図15に基づいて説明する。図14は物理コンフィグレーション編集手段の内部構成を示したものである。図において、物理コンフィグレーション一時記憶手段32は編集中に物理コンフィグレーションを保持する記憶手段であり、表示・表示手段33の行った編集結果は直接物理コンフィグレーション一時記憶手段32に反映される。物理コンフィグレーション一貫性検査手段34は物理コンフィグレーション一時記憶手段32の内容を論理実行環境定義、論理コンフィグレーション、実行環境マッピング情報と照合して一貫性を検査し、矛盾があればこれを表示・編集手段33に伝える。一貫性の検査は、例えば図15に示すアルゴリズムにより実現することができる。

【0054】本実施例によれば、物理コンフィグレーション編集手段30に物理コンフィグレーション一貫性検査手段34を備えたことにより、編集作業結果に対する確認チェックを実施することができ、ここで矛盾が検出された場合にはアプリケーション開発者に警告を発するようにしたので、システムの実稼動に先き立ち事前に編集内容の正当性を確認できシステムの健全性を保証することができる。

【0055】実施例6. この発明の第6の実施例について、図16に基づいて説明する。図16は論理実行環境定義の例を示したものであり、論理実行環境は木構造により階層化され、木構造の一つの節に論理リソースの集合が対応付けられる。この例では、論理実行環境の木構造はアプリケーション利用者の組織構造と対応付けられており、木構造の節は各組織単位（企業でいえば部、課、係など）に相当する。各論理リソースは各組織単位に配置され、システムの機能を分担する想定上の計算機を表す。この例では「営業サーバ」「共通サーバ」「倉庫クライアント」などが論理リソースに相当する。

【0056】本実施例によれば、論理実行環境を木構造として定義して管理するようにしたので、論理実行環境を実運用組織体系に即して自然、且つ柔軟に定義、変更することができる。

【0057】実施例7. この発明の第7の実施例を図17について説明する。図17は論理ドメインの例を示したものであり、論理ドメインは図17における論理コンフィグレーションの木構造の節点と一対一に対応づけられて構成されている。木構造のある特定の節点によって表現された論理ドメインは、指定された特定の節点を根とした部分木を構成している全節点と関連づけられた論理リソースの集合の総和として定義される。

【0058】また、図17においては、木構造の節点に節点名を与えている。図で示すように「営業」「倉庫」の様に重複した節点名を与えた場合でも、根となる節点

から該節点までの経路を経路上の各節点の節点名を列挙して得られる表現、例えば「社内・支社B・営業」を節点を一意に識別する識別名として用いることができる。上記の識別名を論理ドメインの識別名として使用することにより、図9に示した論理ドメイン編集手段80、論理ドメイン記憶手段81を省略し、代わりに論理実行環境編集手段50を用いて各節点に節点名の指定機能を付加した構成が可能になる。

【0059】本実施例によれば、論理実行環境を構成している木構造の節点を論理ドメインと1対1に対応付けることで、ドメインの定義を一層容易に行うことができるとともに、アプリケーション開発者の作業をも軽減することができる。

【0060】

【発明の効果】この発明に係る分散アプリケーション開発支援装置は、以上説明したようにして構成されているので、以下に記載されるような効果を奏する。

【0061】この発明によれば、個々のアプリケーション構成要素の配置を物理実行環境とは独立した論理実行環境に対して指定するようにしたので、物理実行環境が変化した場合においても、システムの構築・保守を容易に行うことができる。

【0062】またこの発明によれば、分散システム上の各物理リソース上に配置の必要な実行モジュールを自動的に各物理リソースに配布し、インストールするようにしたので、アプリケーションの配布・インストール作業が容易になる。

【0063】またこの発明によれば、各サービスを呼び出すことのできる範囲をドメインとして指定し、これに基づいて物理実行環境に対して適切な透過性制御を設定するようにしたので、各サービスに対する透過性を正しく制御することができる。また、物理実行環境の変動に対して独立性の高い論理ドメイン定義情報を用いるようにしたので、大規模システムにおいてもサービスの透過性に対する設定・変更作業を柔軟、且つ容易に行うことができる。

【0064】またこの発明によれば、物理コンフィグレーション編集手段を用いて自動生成された物理コンフィグレーション情報をアプリケーション開発者が修正できるようにしたので、物理実行環境にあわせてアプリケーション構成要素の配置を最適化し、実行効率を高めることができる。

【0065】またこの発明によれば、物理コンフィグレーション情報が論理実行環境、および実行環境マッピング情報と矛盾しないように確認チェックできるようにしたので、実稼動に先だって物理コンフィグレーション情報の正当性が確認でき、システムの健全性を保証することができる。

【0066】さらにこの発明によれば、論理実行環境を木構造として管理運用するようにしたので、実運用体系

に即して論理実行環境を柔軟に定義し、また変更することができる。

【0067】加えて、この発明によれば、論理実行環境を構成している木構造の節点を論理ドメインと1対1に対応付けることで、ドメインの定義を容易に行うことができる。

【図面の簡単な説明】

【図1】 この発明の第1の実施例における分散アプリケーション開発支援装置の構成図である。

【図2】 この発明の第1の実施例の分散アプリケーション開発支援装置におけるアプリケーション開発の手順を示した図である。

【図3】 この発明の第1の実施例の分散アプリケーション開発支援装置における論理実行環境編集手段の画面表示例である。

【図4】 この発明の第1の実施例の分散アプリケーション開発支援装置における論理コンフィグレーション編集手段の画面表示例である。

【図5】 この発明の第1の実施例の分散アプリケーション開発支援装置における物理実行環境編集手段の画面表示例である。

【図6】 この発明の第1の実施例の分散アプリケーション開発支援装置における実行環境マッピングの例である。

【図7】 この発明の第1の実施例の分散アプリケーション開発支援装置における物理コンフィグレーション自動生成手段の動作を示した流れ図である。

【図8】 この発明の第2の実施例の分散アプリケーション開発支援装置において、第1の実施例の構成図に対する追加部分を示した図である。

【図9】 この発明の実施例3の分散アプリケーション開発支援装置の構成図である。

【図10】 この発明の第3の実施例の分散アプリケーション開発支援装置における論理ドメイン編集手段の画面表示例である。

【図11】 この発明の第3の実施例の分散アプリケーション開発支援装置におけるサービスのドメイン指定の画面表示例である。

【図12】 この発明の第3の実施例の分散アプリケーション開発支援装置の物理ドメイン設定手段における物

理ドメインの算出手段を示した流れ図である。

【図13】 この発明の第4の実施例における分散アプリケーション開発支援装置の構成図である。

【図14】 この発明の第5の実施例の分散アプリケーション開発支援装置における物理コンフィグレーション編集手段の構成図である。

【図15】 この発明の第5の実施例の分散アプリケーション開発支援装置における物理コンフィグレーション一貫性検査手段の動作を示した流れ図である。

【図16】 この発明の第6の実施例6の分散アプリケーション開発支援装置における論理実行環境の例を示した図である。

【図17】 この発明の実施例7の分散アプリケーション開発支援装置における論理ドメインの例を示した図である。

【図18】 従来の分散アプリケーション開発支援装置の構成図である。

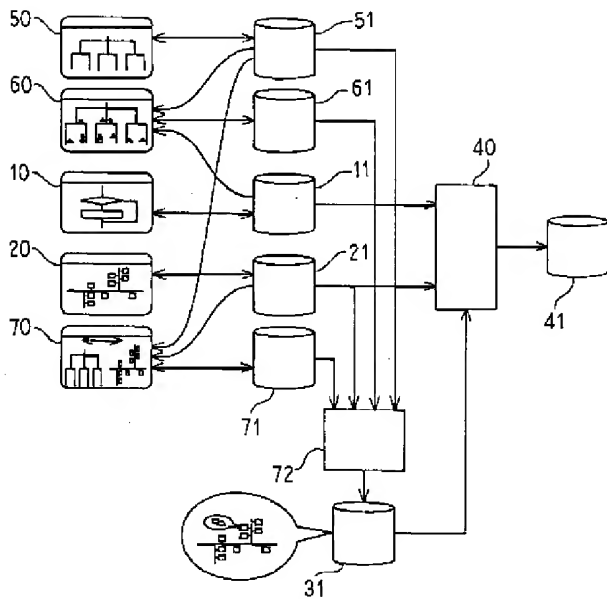
【図19】 従来の分散アプリケーション開発支援装置におけるアプリケーション開発の手順である。

【図20】 従来の分散アプリケーション開発支援装置における物理コンフィグレーション編集手段の画面表示例である。

【符号の説明】

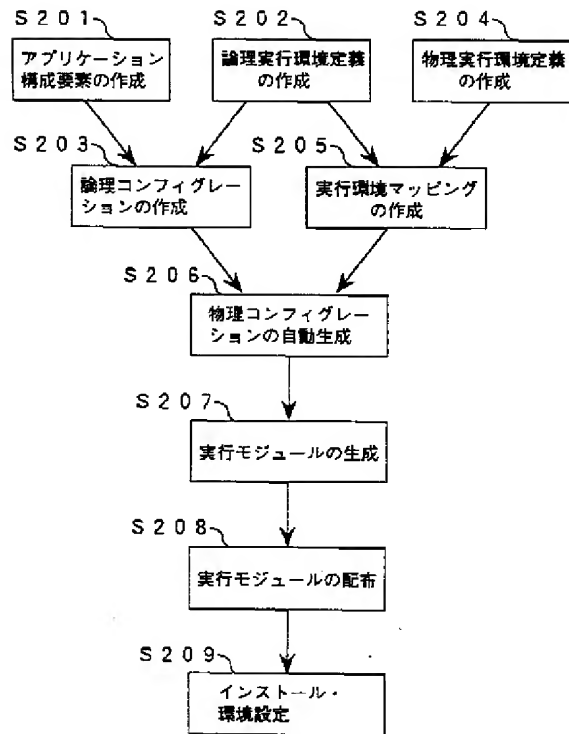
10 アプリケーション構成要素定義手段、11 アプリケーション構成要素記憶手段、20 物理実行環境編集手段、21 物理実行環境記憶手段、30 物理コンフィグレーション編集手段、31 物理コンフィグレーション記憶手段、32 物理コンフィグレーション一時記憶手段、33 物理コンフィグレーション表示・編集手段、34 物理コンフィグレーション一貫性検査手段、40 実行モジュール生成手段、41 実行モジュール記憶手段、42 アプリケーションインストール手段、43 物理ドメイン算出手段、44 物理ドメイン設定手段、50 論理実行環境編集手段、51 論理実行環境記憶手段、60 論理コンフィグレーション編集手段、61 論理コンフィグレーション記憶手段、70 実行環境マッピング編集手段、71 実行環境マッピング記憶手段、72 物理コンフィグレーション自動生成手段、80 論理ドメイン編集手段、81 論理ドメイン記憶手段。

【図 1】



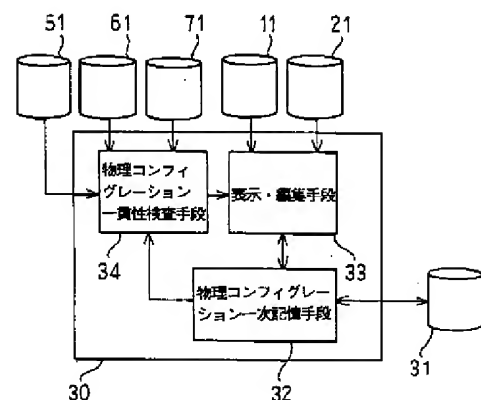
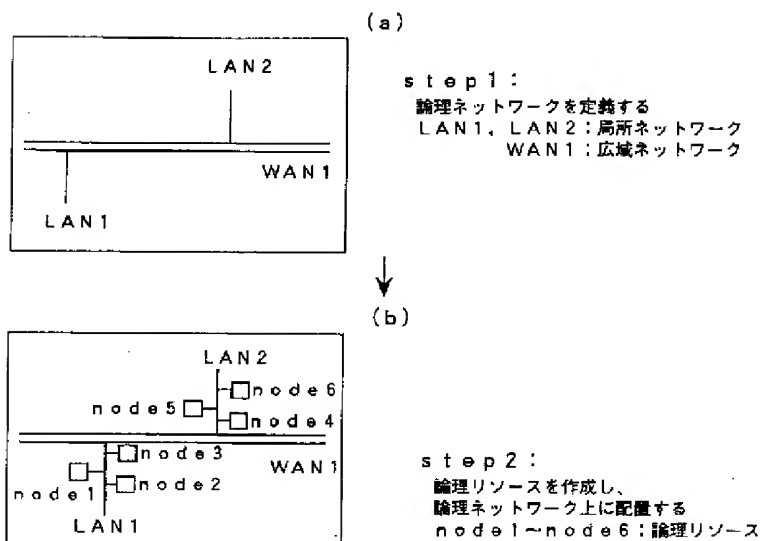
10:アプリケーション構成要素定義手段 50:論理実行環境編集手段  
 11:アプリケーション構成要素記憶手段 51:論理実行環境記憶手段  
 20:物理実行環境編集手段 60:論理コンフィグレーション編集手段  
 21:物理実行環境記憶手段 61:論理コンフィグレーション記憶手段  
 31:物理コンフィグレーション記憶手段 70:実行環境マッピング編集手段  
 40:実行モジュール生成手段 71:実行環境マッピング記憶手段  
 41:実行モジュール記憶手段 72:物理コンフィグレーション自動生成手段

【図 2】



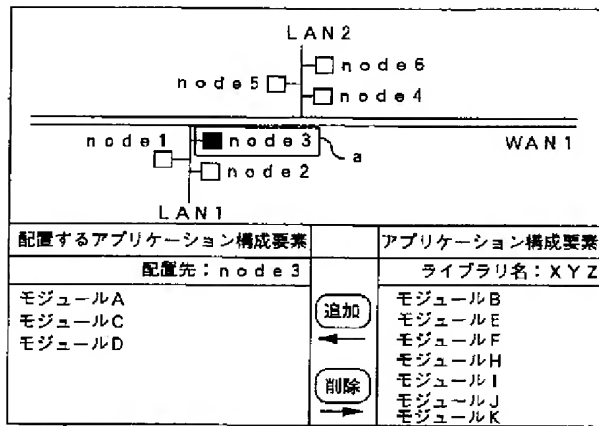
【図 14】

【図 3】



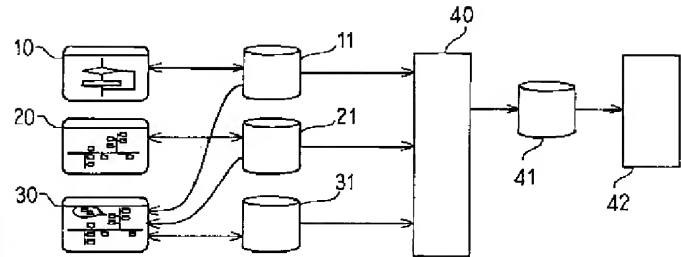
11:アプリケーション構成要素記憶手段  
 21:物理実行環境記憶手段  
 30:物理コンフィグレーション編集手段  
 31:物理コンフィグレーション記憶手段  
 51:論理実行環境記憶手段  
 61:論理コンフィグレーション記憶手段  
 71:実行環境マッピング記憶手段

【図4】



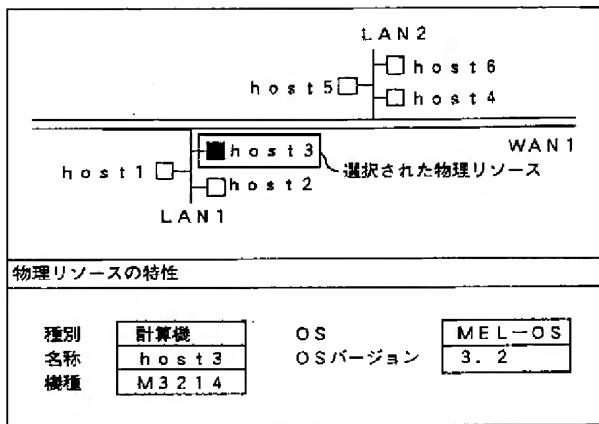
- a 配置先の論理ソース  
 b 選択された論理ソース上に配置するアプリケーション構成要素のリスト  
 c アプリケーション構成要素のリスト

【図18】



- 10: アプリケーション構成要素定義手段 40: 実行モジュール生成手段  
 11: アプリケーション構成要素記憶手段 41: 実行モジュール記憶手段  
 20: 物理実行環境編集手段 42: アプリケーションインストール手段  
 21: 物理実行環境記憶手段  
 30: 物理コンフィグレーション編集手段  
 31: 物理コンフィグレーション記憶手段

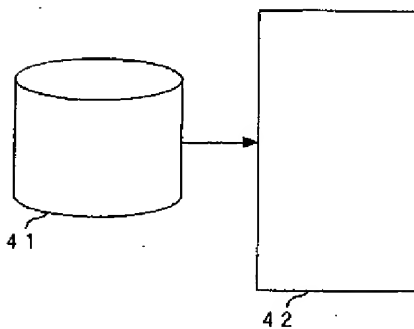
【図5】



物理リソースの特性の定義

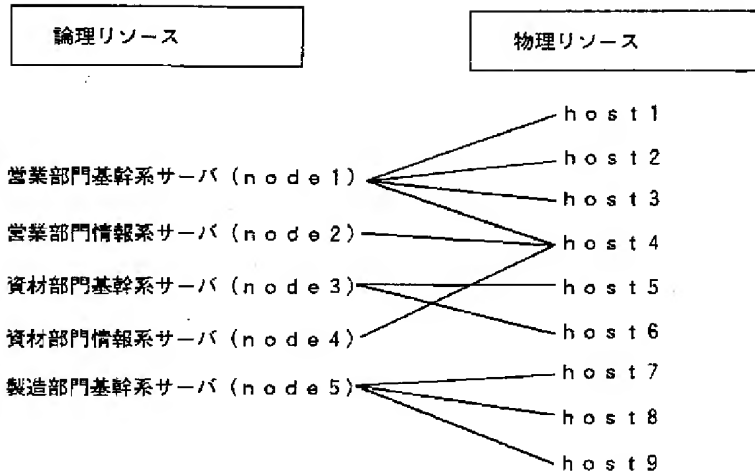
host 1 ~ host 6; 物理リソース

【図8】

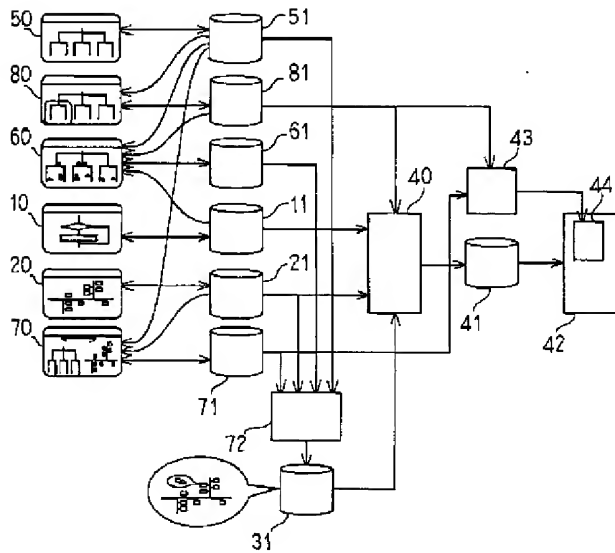


- 41: 実行モジュール記憶手段  
 42: アプリケーションインストール手段

【図6】



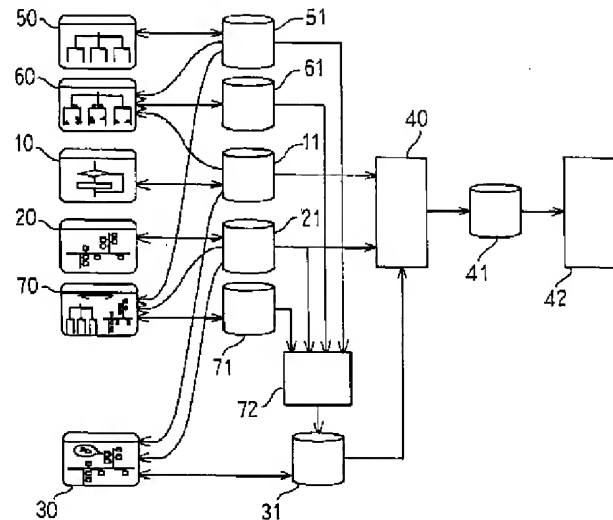
【図9】



10:アプリケーション構成要素定義手段  
11:アプリケーション構成要素記憶手段  
20:物理実行環境編集手段  
21:物理実行環境記憶手段  
31:物理コンフィグレーション記憶手段  
40:実行モジュール生成手段  
41:実行モジュール記憶手段  
42:アプリケーションインストール手段  
43:物理ドメイン算出手段  
44:物理ドメイン設定手段

50:論理実行環境編集手段  
51:論理実行環境記憶手段  
60:論理コンフィグレーション編集手段  
61:論理コンフィグレーション記憶手段  
70:実行環境マッピング編集手段  
71:実行環境マッピング記憶手段  
72:物理コンフィグレーション自動生成手段  
80:論理ドメイン編集手段  
81:論理ドメイン記憶手段

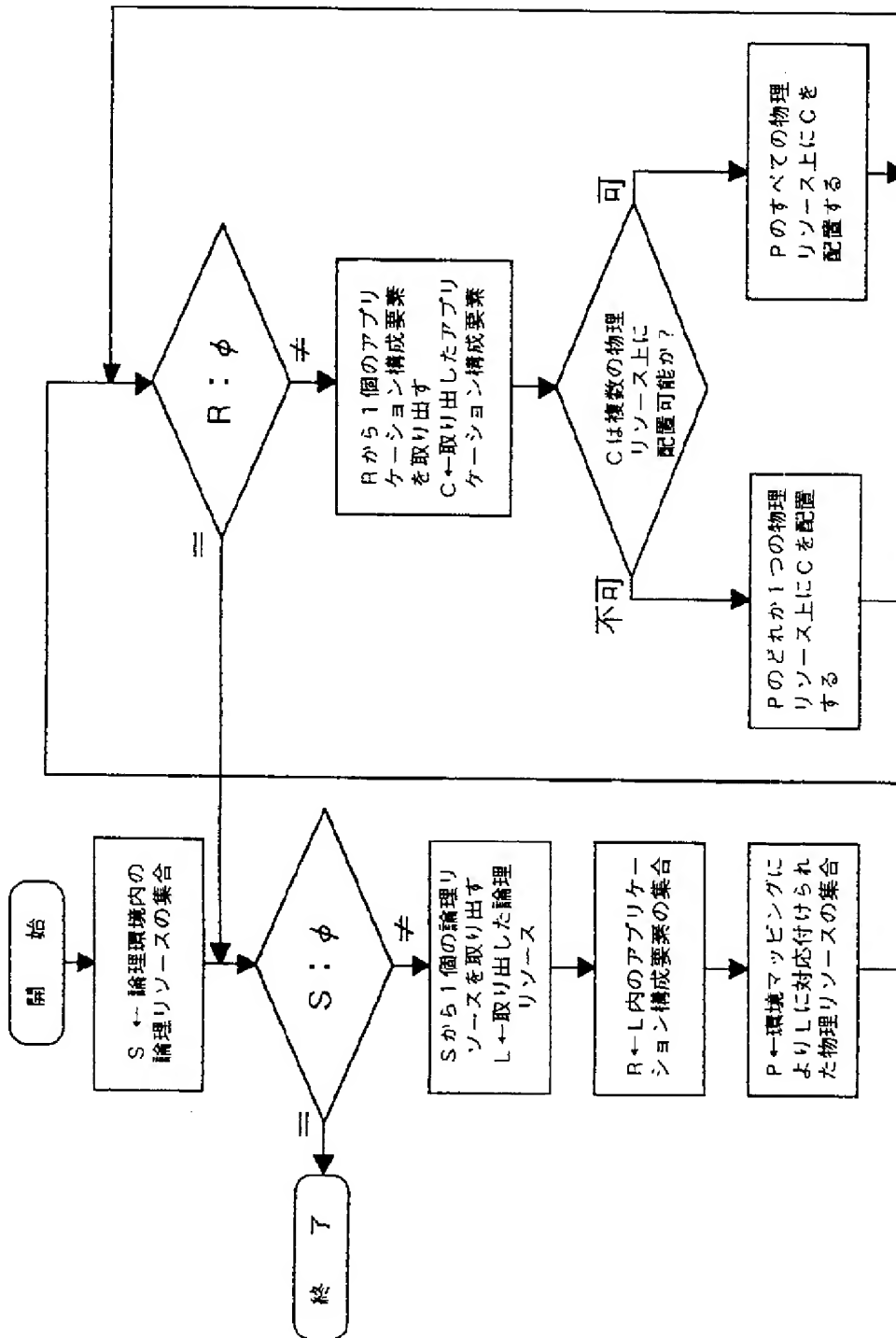
【図13】



10:アプリケーション構成要素定義手段  
11:アプリケーション構成要素記憶手段  
20:物理実行環境編集手段  
21:物理実行環境記憶手段  
30:物理コンフィグレーション編集手段  
31:物理コンフィグレーション記憶手段  
40:実行モジュール生成手段  
41:実行モジュール記憶手段  
42:アプリケーションインストール手段

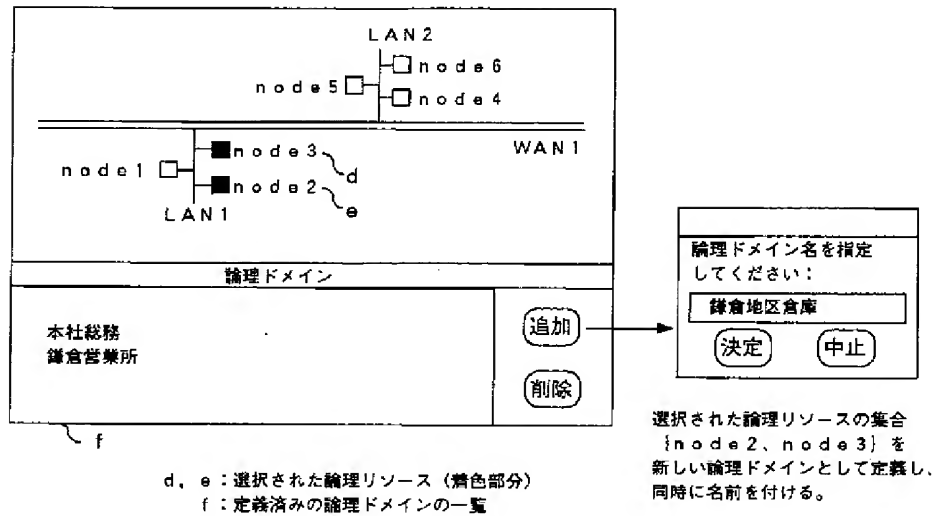
50:論理実行環境編集手段  
51:論理実行環境記憶手段  
60:論理コンフィグレーション編集手段  
61:論理コンフィグレーション記憶手段  
70:実行環境マッピング編集手段  
71:実行環境マッピング記憶手段  
72:物理コンフィグレーション自動生成手段

【図7】

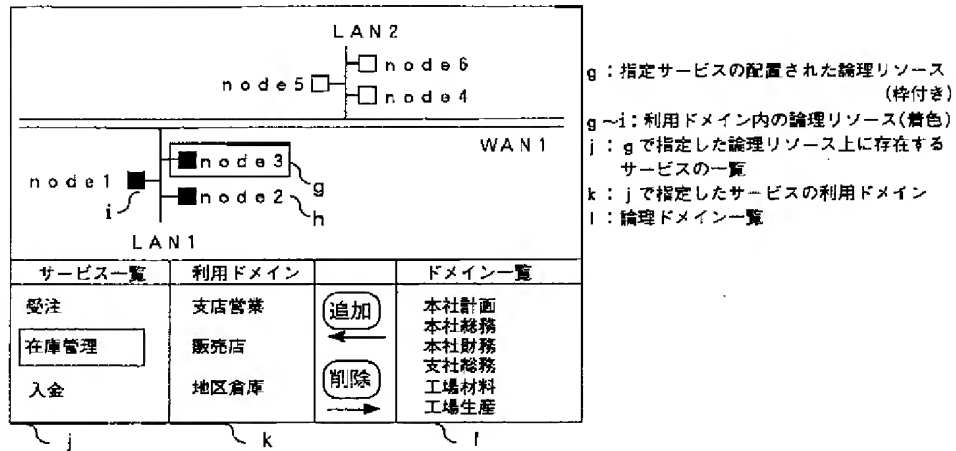




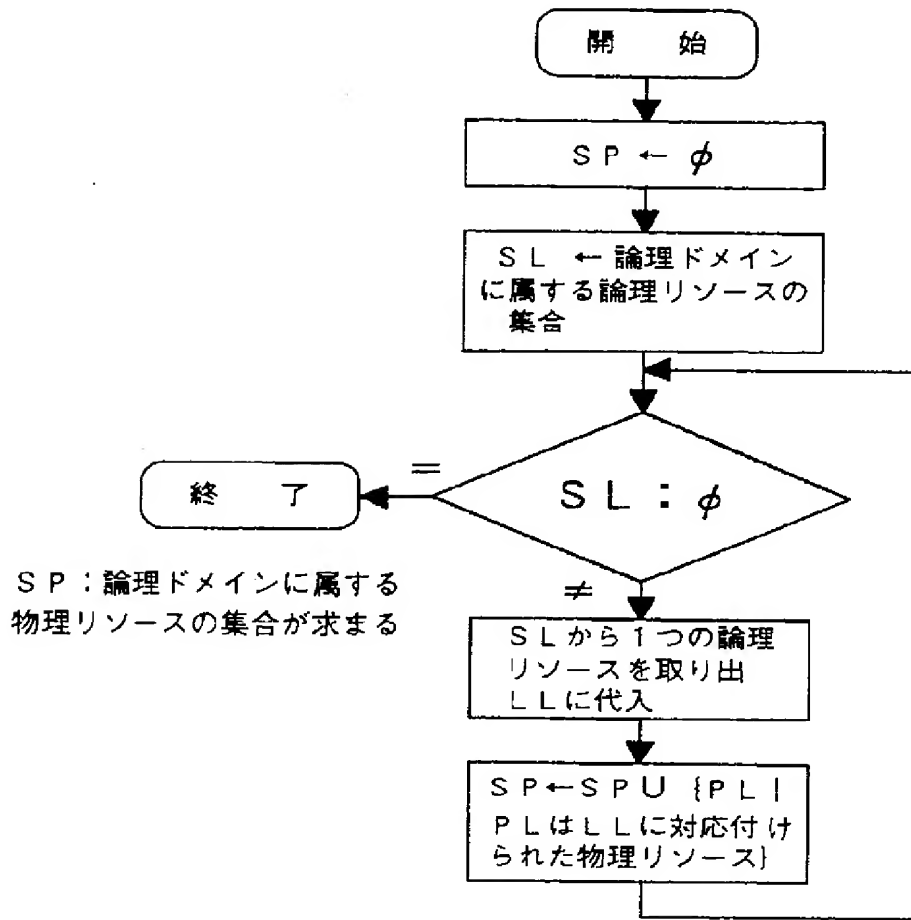
【図10】



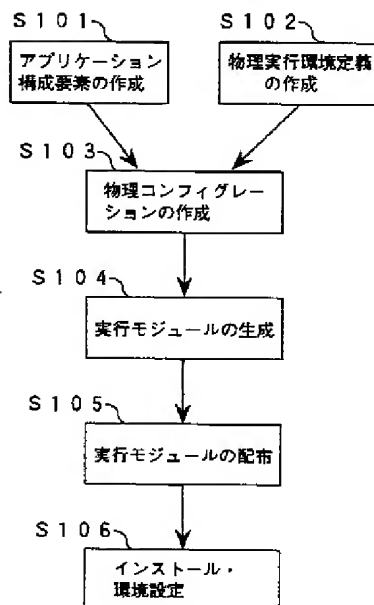
【図11】



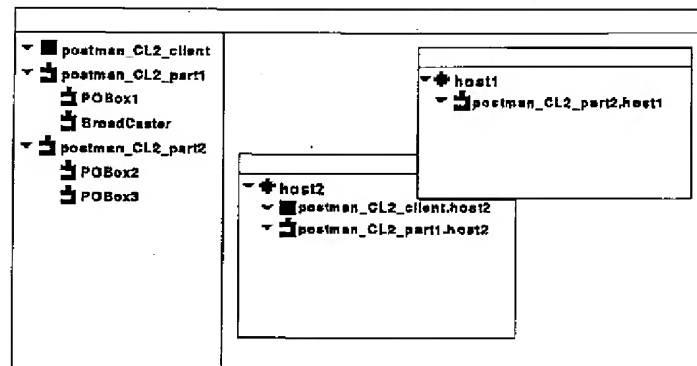
【図12】



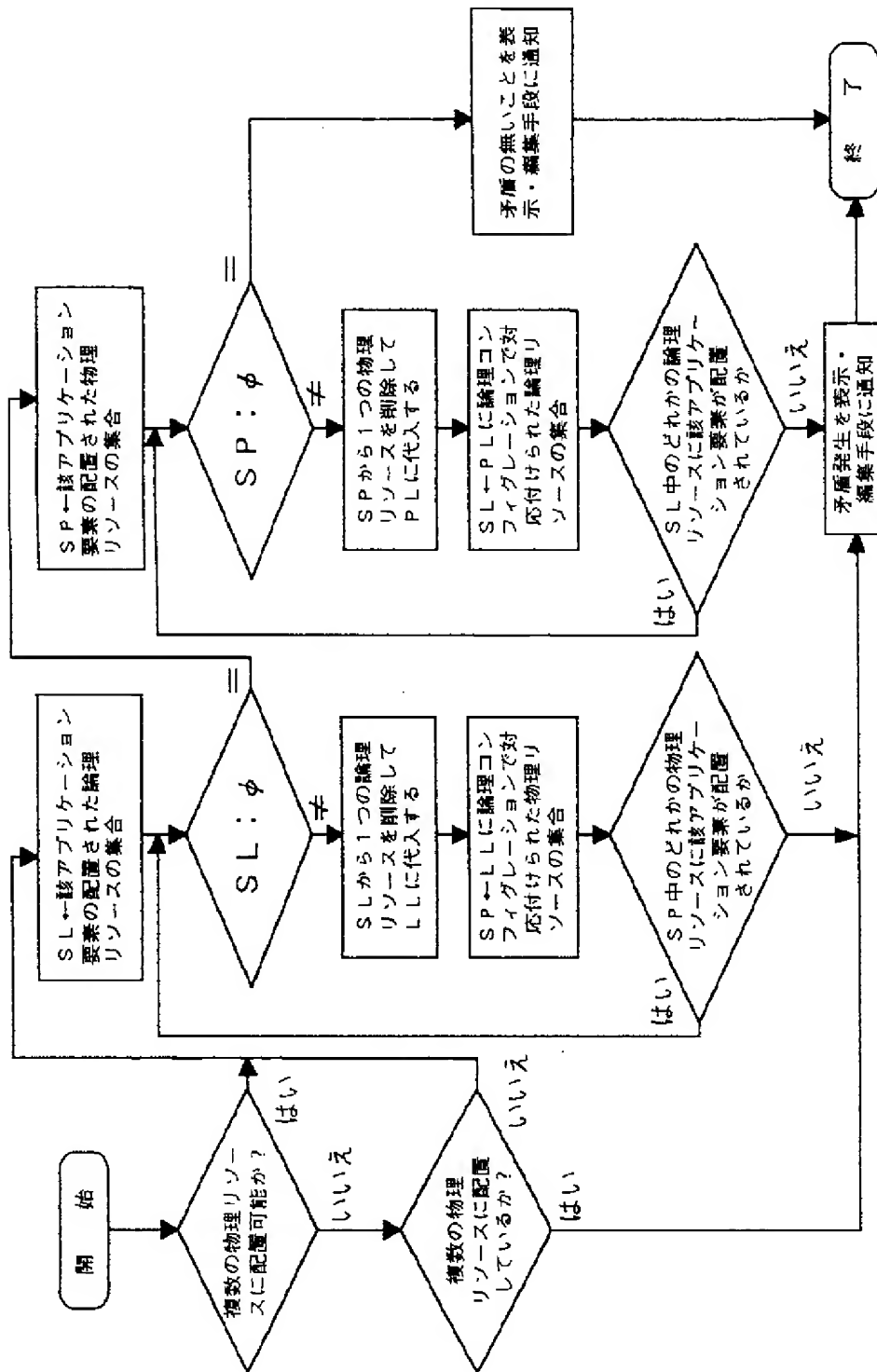
【図19】



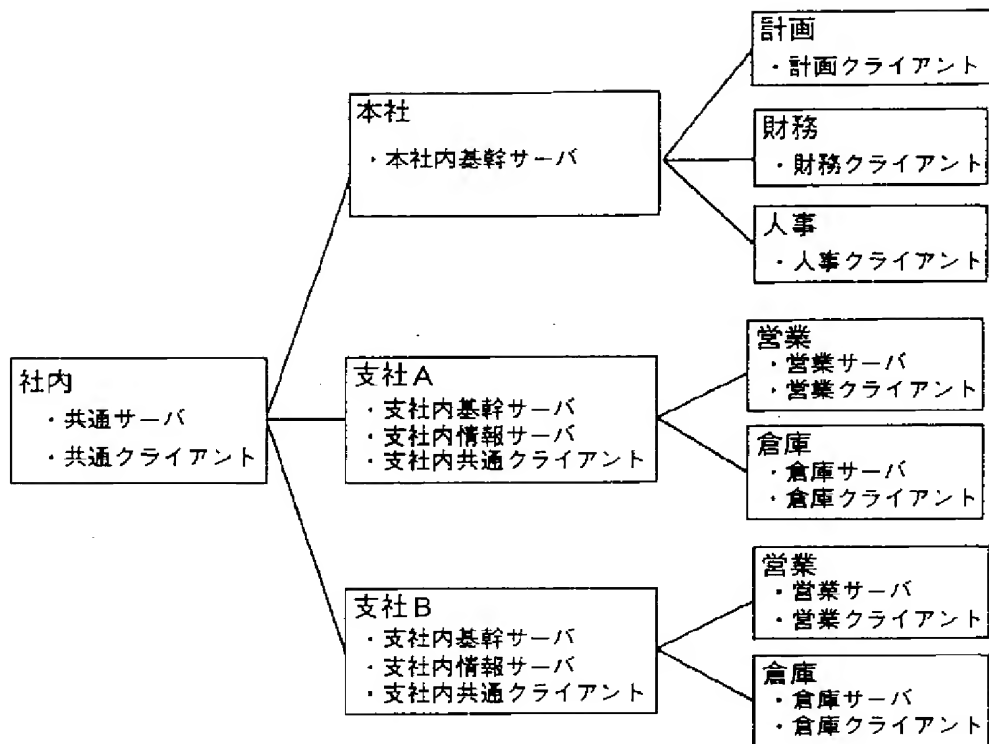
【図20】



【図15】



【図16】



【図17】

